A Set-oriented MOEA/D

Bilel Derbel
Univ. Lille, CRIStAL
Inria Lille – Nord Europe
Lille, France
bilel.derbel@univ-lille1.fr

Arnaud Liefooghe
Univ. Lille, CRIStAL
Inria Lille – Nord Europe
Lille, France
arnaud.liefooghe@univ-lille.fr

Qingfu Zhang
The City University of Hong Kong
Shenzhen Research Institute
Hong Kong, Shenzhen, China
qingfu.zhang@cityu.edu.hk

Sébastien Verel
Université du Littoral Côte
Calais, France
verel@univ-littoral.fr

Hernán Aguirre
Shinshu University
Nagano, Japan
ahernan@shinshu-u.ac.jp

Kiyoshi Tanaka
Shinshu University
Nagano, Japan
ktanaka@shinshu-u.ac.jp

ABSTRACT

The working principles of the well-established multi-objective evolutionary algorithm MOEA/D relies on the iterative and cooperative improvement of a number of single-objective sub-problems obtained by decomposition. Besides the definition of sub-problems, selection and replacement are, like in any evolutionary algorithm, the two core elements of MOEA/D. We argue that these two components are however loosely coupled with the maintained population. Thereby, we propose to re-design the working principles of MOEA/D by adopting a set-oriented perspective, where a many-to-one mapping between sub-problems and solutions is considered. Selection is then performed by defining a neighborhood relation among solutions in the population set, depending on the corresponding sub-problem mapping. Replacement is performed following an elitist mechanism allowing the population to have a variable, but bounded, cardinality during the search process. By conducting a comprehensive empirical analysis on a range of combinatorial multi- and many-objective NK-landscapes, we show that the proposed approach leads to significant improvements, especially when dealing with an increasing number of objectives. Our findings indicate that a set-oriented design can constitute a sound alternative for strengthening the practice of multi- and many-objective evolutionary optimization based on decomposition.

CCS CONCEPTS

• **Theory of computation → Evolutionary algorithms**; **Random search heuristics**; • **Computing methodologies → Optimization algorithms**; **Discrete space search**; • **Applied computing → Multi-criterion optimization and decision-making**;

KEYWORDS

multi- and many-objective optimization, decomposition, evolutionary algorithms, combinatorial optimization

ACM Reference Format:
Bilel Derbel, Arnaud Liefooghe, Qingfu Zhang, Sébastien Verel, Hernán Aguirre, and Kiyoshi Tanaka. 2018. A Set-oriented MOEA/D. In *GECCO '18: Genetic and Evolutionary Computation Conference, July 15–19, 2018, Kyoto, Japan.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3205455.3205575

GECCO '18, July 15–19, 2018, Kyoto, Japan
2018. ACM ISBN 978-1-4503-5618-3/18/07.
https://doi.org/10.1145/3205455.3205575

1 INTRODUCTION

Context. A multi-objective problem requires to optimize multiple objectives simultaneously. Since the objectives are many-often conflicting, it is unlikely to find one single solution optimizing all target objectives. One of the most challenging issues is then to find a set of solutions providing different trade-offs between the considered objectives. For this purpose, Evolutionary Multi-objective Optimization (EMO) algorithms have been proved to be extremely efficient in finding high-quality approximation sets [3, 19].

Most existing EMO approches follow a standard algorithmic scheme where a *population* of solutions is evolved in an iterative manner. Specific *mating selection* and *replacement* mechanisms are used to respectively (i) choose some parents from which new individuals are created by means of variation operators, and (ii) update the population by incorporating the newly-generated offspring solutions. Apart from the variation operators, it is well understood that the dynamics of an evolutionary search process in terms of convergence and diversity is tightly related to the design of the selection and replacement mechanisms. Different classes of EMO algorithms can be distinguished according to these two critically-important components. For instance, dominance-based approaches, such as NSGA-II [3] and PLS [15], rely on a dominance relation for "comparing" different solutions at the selection and replacement steps. Indicator-based approaches, such as IBEA [20] and SMS-MOEA [2], rely on a quality indicator to guide the evolutionary search process when deciding the most suitable parents to select and the most suitable solution to maintain in the population. In this paper, we are interested in aggregation-based algorithms, e.g., [4, 5, 8, 14, 19].

Motivations. Our main focus is on the design principles of the so-called MOEA/D (Multi-objective evolutionary algorithm based on decomposition) algorithm [8, 19], which can be considered as a state-of-the-art framework in the class of aggregation-based approaches. In its basic variant, given a scalarizing function in the objective space, MOEA/D considers the cooperative solving of different single-objective sub-problems, each one defined using the given scalarizing function parametrized by a different weight vector. A population of solutions is then considered, where each sub-problem, equivalently weight vector or search direction, is mapped to one single solution.

The computational flow of MOEA/D consists in iterating over the set of weight vectors, by performing selection and replacement in the *vicinity* of the current weight vector. In fact, MOEA/D does *not* iterate *explicitly* over the solutions when performing selection and

replacement, which might raise some concerns. For example, when the number and distribution of the weight vectors does not properly match the distribution of non-dominated solutions, MOEA/D may imply a population of multiple duplicates of the same solution. This might not only impact the (small) cardinality of the output solution set and a lack of diversity in the current population, it also biases the selection of the crucially-important mating selection mechanisms. By handling a *set* of solutions, where duplicates are *not* allowed, it can be possible to offer an alternative (re-)design of MOEA/D. This idea is the core motivation of our work. In other words, a notable design feature of MOEA/D, to be contrasted with the previously-mentioned EMO algorithms, is that the population is *not* controlled in an explicit manner. We claim that this plays an important role in the behavior of MOEA/D, and subsequently in the performance of the underlying evolutionary search process. Besides, we argue that investigating the impact of such a design feature shall allow us to better grasp some of the common issues when choosing MOEA/D as an effective solving approach.

Contributions and Results Overview. In this paper, we adopt a set-oriented perspective allowing us to design an enhanced EMO algorithm based on decomposition. We propose to explicitly evolve a population of solutions considered as a *bounded set*. We then manage to loosely couple the sub-problems solving with the iterations of the evolutionary process, while still using the underlying (fixed-size) weight vectors as a core element to perform selection and replacement. Based on the scalarized fitness values of solutions, we adopt a many-to-one mapping of the weight vectors to the population, i.e. a subset of weight vectors can be explicitly assigned to one single solution. Mating selection is performed within the neighborhood of solutions, which is defined using the distance in the objective space between the corresponding subsets of weight vectors to which the solutions are mapped. Replacement is performed following a standard elitist mechanism where the population size is not fixed, but bounded by the number of weight vectors.

To validate these design principles, we experiment the proposed algorithm using ρMNK-landscapes, considered as a standard generic combinatorial optimization problem family, with variable problem size, number of objectives, and objective correlation. Through a comprehensive empirical analysis, the proposed algorithm is shown to improve the quality of the obtained approximation sets, and to have a robust behavior in a relatively wide range of problems and configurations. In particular, a substantial improvement is reported as the number of objectives increases and as the maximum size of the population scales. More importantly, our investigations provide evidence that a set-oriented design offers an alternative perspective to the establishment of future high-quality decomposition-based EMO algorithms for both multi- and many-objective optimization.

Outline. The paper is organized as follows. Section 2 presents some necessary background and related work. Section 3 introduces the proposed set-oriented MOEA/D framework. Section 4 provides the experimental analysis of our approach. The last section concludes the paper and discusses open issues.

2 BACKGROUND AND RELATED WORK

2.1 Multi-objective Optimization

A *multi-objective optimization problem* (MOP) can be defined by a set of M objective functions $f = (f_1, f_2, \ldots, f_M)$, and a set X of feasible solutions in the *decision space*. In the combinatorial case, X is a discrete set. Let $Z = f(X) \subseteq \mathbb{R}^M$ be the set of feasible outcome vectors in the *objective space*. In a maximization context, an objective vector $z \in Z$ is *dominated* by a vector $z' \in Z$ ($z \prec z'$) iff $\forall m \in \{1, \ldots, M\}$, $z_m \leqslant z'_m$ and $\exists m \in \{1, \ldots, M\}$ s.t. $z_m < z'_m$. A solution $x \in X$ is dominated by a solution $x' \in X$ ($x \prec x'$) iff $f(x) \prec f(x')$. A solution $x^\star \in X$ is *Pareto optimal* if there does not exist any other solution $x \in X$ such that $x^\star \prec x$. The set of all Pareto optimal solutions is the *Pareto set*. Its mapping in the objective space is the *Pareto front*.

2.2 The MOEA/D Framework

Decomposition-based EMO algorithms [4, 5, 19] seek good-performing solutions in multiple regions of the Pareto front by *decomposing* the original MOP into a number of *scalarized* single-objective *sub-problems* [12]. In this paper, we use the weighted Chebyshev (g^{te}) scalarizing function, to be minimized:

$$g^{\text{te}}(x, \lambda) = \max_{i \in \{1, \ldots, M\}} \lambda_i \cdot \left| z_i^\star - f_i(x) \right|$$

where $x \in X$, $\lambda = (\lambda_1, \ldots, \lambda_M)$ is a positive weight vector such that $\lambda_i \geqslant 0$ for all i, and $z^\star = (z_1^\star, \ldots, z_M^\star)$ is a reference point such that, $\forall i \in \{1, \ldots, M\}, \forall x \in X, z_i^\star > f_i(x)$.

In MOEA/D [8, 19], sub-problems are optimized cooperatively by defining a *neighborhood relation* between sub-problems. Given a set of μ weight vectors $\Lambda_\mu = (\lambda^1, \ldots, \lambda^\mu)$ defining μ sub-problems, MOEA/D maintains a population $P_\mu = (x^1, \ldots, x^\mu)$ where each individual corresponds to one sub-problem. For each sub-problem $i \in \{1, \ldots, \mu\}$, a set of neighbors $\mathcal{B}(i)$ is defined by considering the T-closest weight vectors using the euclidian distance (denoted $|| \cdot ||$). The sub-problems are optimized iteratively. For each sub-problem i, two sub-problems are selected at random from $\mathcal{B}(i)$, and the two corresponding solutions are considered as parents. An offspring x' is created by means of variation operators (e.g., mutation, crossover). For every $j \in \mathcal{B}(i)$, if x' improves j's current solution x^j, then x' replaces it, i.e., if $g^{\text{te}}(x', \lambda^j) < g^{\text{te}}(x^j, \lambda^j)$ then $x^j = x'$. The algorithm loops over the sub-problems, i.e., weight vectors, until a stopping condition is satisfied.

2.3 Discussion and Positioning

Rationale. Since mating selection and replacement in MOEA/D are performed among neighbors, the neighborhood definition, $\mathcal{B}(i)$, is critical for an accurate diversity/convergence balance. For example, replacement can lead to the situation where a solution which improves a sub-problem that does not belong to the neighborhood, is discarded. It can also lead to a situation where several neighbors are assigned the same solution. This has the effect of decreasing the probability that different solutions are picked at the selection step, since the T different neighbors of a sub-problem, in the weight vector space, could imply much fewer than T different solutions in the search space. Such a feature can prevent the reproduction operator from producing high-quality offsprings. This might not only slow

down convergence, but it might also make the search process stuck more easily. Additionally, since the MOEA/D main iterative process is performed w.r.t. sub-problems, if the solutions maintained by multiple sub-problems are the same, then the corresponding genotype is likely to be considered more often for reproduction. This might not be an issue at a first sight, especially if such a scenario happened because an exceptionally-good solution w.r.t respect to a subset of sub-problems was found at some iteration. However, this might be unfair w.r.t some other sub-problems, because their solutions are then less likely to be considered for reproduction.

Related Works. Different techniques can be adopted to deal with such issues, e.g., adapting the neighborhood size, adjusting the neighborhood w.r.t. the selected sub-problem, restricting the number of replacements, see [16]. With respect to the population management, different variants can also be found, for instance by adopting a global (generational) replacement approach [10, 18] or a stable matching methodology [9]. One variant, denoted MOEA/D$_{(\delta,nr)}$, was extensively considered in the literature [8]. Two modifications are considered. The first one consists in an extra probability parameter δ allowing parents to be selected from the whole population instead of solely the T-neighborhood. The second one consists in an extra parameter nr bounding the number of neighbors that can be replaced by a newly-generated offspring.

Positioning. To the best of our knowledge, there are no much investigations addressing the impact of iterating over the weight vectors Λ_μ in MOEA/D, instead of explicitly iterating over the population P_μ, considered as a solution *set*. One can only find few recent works aiming at understanding the behavior of the population dynamics in MOEA/D [13]. It is in fact worth-noticing that our ultimate goal is not to provide yet another variant of MOEA/D, but rather to better grasp the complexity of solving a multi-objective optimization problem by evolving a population based on the concept of decomposition. In this respect, the community still lacks much understanding on the subject despite the success of the MOEA/D framework. The population maintained in MOEA/D cannot be really considered as a set, in particular because it could include duplicates. Consequently, in contrast to other classes of algorithms, the behavior of MOEA/D cannot be easily understood from a set-oriented multi-objective search perspective [21], which is known to provide a unified theory and best practices of EMO algorithms.

Viewing MOEA/D from a set-oriented angle requires to revise its initial design components. In particular, this requires to re-design its main iterative process, as well as the underlying selection and replacement mechanisms, while keeping the spirit of the ingredients that made its wide success, that is: (i) the use of decomposition and (ii) the cooperation among sub-problems. The rest of this paper is devoted to showing how such a perspective can offer new alternatives to enhance the performance of MOEA/D while opening a new angle for designing EMO algorithms based on decomposition.

3 A SET-ORIENTED DESIGN OF MOEA/D

In the following, we adopt a set-oriented perspective to the design of MOEA/D. We still consider μ weight vectors defining μ sub-problems, for which we search for a *set* of good performing solutions. However, we consider the population as a bounded *set*, which is managed in a more explicit manner. The general idea is to map the weight vectors to the solutions, and to use the mapping information to perform selection and replacement following a standard evolutionary scheme. The flow of the proposed scheme, called SET-MOEA/D, is illustrated in Fig. 1, and detailed in Algorithm 1.

Population Mapping. Given the *current* population P, we compute *at every iteration* a mapping $\phi : \Lambda_\mu \overset{g}{\longmapsto} P$ of the target μ sub-problems into the solutions of the population. We adopt an elitist mapping function where every weight vector $\lambda^i \in \Lambda_\mu$ is mapped to exactly one solution $x \in P$ having the best fitness value w.r.t. the scalarizing function g (lines 4 to 6). Since different sub-problems can have the *same* best solution, ϕ is a *many-to-one* mapping where a solution $x \in P$ can be either: (i) mapped by one or more weight vectors, denoted $\Phi(x)$ in line 6, or (ii) non-mapped by any weight vector, i.e., $\Phi(x) = \varnothing$. The non-mapped solutions are discarded from the population (line 8). After this step, all solutions remaining in the population are different, and duplicates cannot survive.

Neighborhood-based Selection. We consider a standard evolutionary (genetic-like) process where two parents are used for reproduction. The first parent $\hat{x}$ is selected uniformly at random from the population (line 9). The second parent $\check{x}$ is however selected in the neighborhood $\mathcal{BS}(\hat{x})$ of the first parent (lines 11 and 12). Unlike the conventional MOEA/D, the neighborhood is here defined explicitly w.r.t a given solution. The distance between two solutions in the population set P (denoted d in line 10) is defined according to the distance between the weight vectors to which they are respectively mapped. In more details, since ϕ is a many-to-one mapping, a solution x surviving in the population can be mapped to a subset of weight vectors $\Phi(x)$. The distance between any two solutions x and x' in the population is then defined as the minimum distance between any pair of weight vectors respectively in $\Phi(x)$ and $\Phi(x')$. Since the population *cannot* contain any duplicates, this selection mechanism ensures that the genotypes of parents are different and likely to be close-enough to produce promising individuals.

Reproduction and Population Update. An offspring solution y is produced (line 13) and injected in the population P (line 14). Nevertheless, this does *not* mean that it will necessarily survive. In fact, as soon as the next iteration starts, the mapping ϕ is computed again, and the population is updated by removing non-mapped solutions. Actually, the mapping enables the replacement in our design. With the newly-generated offspring y, the updated mapping may imply different scenarios. First, y can survive iff it is elected as the best solution for at least one weight vector, i.e., $\Phi(y) \neq \varnothing$. In such a scenario, if a solution x, to which a weight vector in $\Phi(y)$ was mapped in the previous iteration, is still the best for at least one other weight vector, then x survives as well, otherwise it dies. This means that if the offspring y survives, then a number of other solutions might die. In all cases, the population size cannot exceed the pre-defined number of weights μ, but it can drop to less than μ when an offspring improves simultaneously different sub-problems.

Implementation Details and Further Considerations. Notice that the mapping ϕ does *not* need to be recomputed from scratch at each iteration. Since only the mapping of some weights is likely to change at a given iteration w.r.t. a newly-produced offspring y, one

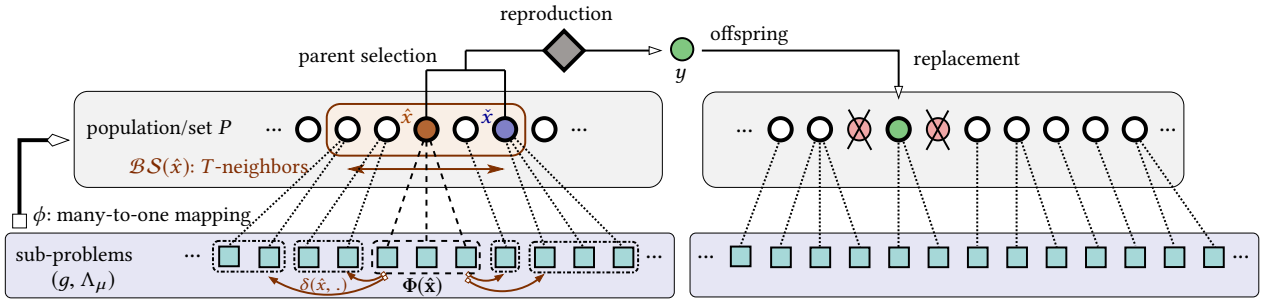

Figure 1: A schematic illustration of SET-MOEA/D: many-to-one mapping ϕ, T-neighbors, mating selection, and replacement.

Algorithm 1: A set oriented MOEA/D (SET-MOEA/D)

Input: $\Lambda_\mu := \{\lambda^1, \ldots, \lambda^i, \ldots, \lambda^\mu\}$: weight vectors;
$g(\cdot \mid \lambda)$: a scalarizing function to be minimized;
T: neighborhood size

$P := \bigcup x \leftarrow$ initial population of solutions ;
while *STOPPING CONDITION* **do**
  `// Replacement: many-to-one mapping function` ϕ `of` μ
  `sub-problems to solution set (`$\Phi := \phi^{-1}$ `)`
  for $x \in P$ **do** $\Phi(x) \leftarrow \varnothing$;
  for $i \in \{1, \ldots, \mu\}$ **do**
    $x \leftarrow \arg\min_{x \in P} g(x \mid \lambda_i)$;
    $\phi(\lambda^i) \leftarrow x$; $\Phi(x) \leftarrow \Phi(x) \cup \{\lambda^i\}$;
  `// Replacement: remove non-mapped solutions`
  for $x \in P$ **do**
    if $\Phi(x) = \varnothing$ **then** $P \leftarrow P \setminus \{x\}$;
  `// Select one parent`
  $\hat{x} \leftarrow$ select a (random) parent in P;
  `// Compute` T`-Neighboring solutions`
  for $x \in P$ **do** $\mathrm{d}(\hat{x}, x) \leftarrow min_{\lambda \in \Phi(\hat{x}), \lambda' \in \Phi(x)} ||\lambda, \lambda'||$; ;
  $\mathcal{BS}(\hat{x}) \leftarrow$ the T closest solutions in P w.r.t. d;
  `// Select another parent`
  $\check{x} \leftarrow$ select a (random) parent in $\mathcal{BS}(\hat{x})$;
  `// Reproduction, e.g. crossover and mutation`
  $y \leftarrow$ generate an offspring from parents $\hat{x}$ and $\check{x}$;
  `// Replacement: merge population with offspring`
  $P \leftarrow P \cup \{y\}$;

simply needs to compute the aggregated fitness values of the newly created offspring, and then update the mapping and the distances required to define $\mathcal{BS}$ in an incremental manner. This can only imply a restricted complexity cost compared to MOEA/D. More importantly, the proposed SET-MOEA/D algorithm is to be viewed as *one* specific implementation of a set-oriented design of MOEA/D. In fact, the underlying algorithmic framework can be *flexibly* extended with other components guided *both* by: (i) decomposition-specific aspects, such as extended notions of neighborhood(s) w.r.t. solutions or other mapping functions, and (ii) by more general plug-and-play EMO techniques, such as steady-state, generational, 'comma' or 'plus' evolutionary strategies. In this respect, decomposition is simply to be considered as a way of efficiently structuring the population, which can then be handled in a more explicit manner.

4 RESULTS AND EXPERIMENTAL ANALYSIS

4.1 Experimental Setup

ρMNK-Landscapes. we use ρMNK-landscapes considered as a family of problem-independent models for multi-objective multi-modal combinatorial optimization [1, 6, 17]. Feasible solutions are as binary strings of size N. For every $k \in \{1, \ldots, M\}$, the k^{th} objective (to be maximized) is defined by: $f_k(x) = \frac{1}{N} \sum_{i=1}^{N} c_i^k(x_i, x_{i_1}, \cdots, x_{i_K})$, such that c_i^k is the component functions. Component-values follow a multivariate uniform distribution of dimension M, defined by a correlation matrix [17]. We consider the same correlation between all pairs of objectives, given by a coefficient $\rho > \frac{-1}{M-1}$. A positive (resp. negative) coefficient decreases (resp. increases) the degree of conflict between the objective values. In our experiments, we set $\rho \in \{-0.2, 0.0, 0.2\}$, $M \in \{2, 3, 4, 5\}$, $N = 100$ and $K = 2$. Investigating other values for K and N are left for future work.

Parameter Setting. SET-MOEA/D, the conventional MOEA/D [19] as well as MOEA/D$_{(\delta, nr)}$ [8], are considered in our experiments. The number of weight vectors μ is set in the range $\{100, 200, 600\}$ and generated using the method described in [11]. Each considered configuration is executed 20 times. The stopping condition is set to 10^6 evaluation function calls. We use an independent bit-flip mutation with a rate $\frac{1}{N}$ and one-point crossover. The neighborhood size T is set to 10% of μ. A population of μ randomly-generated individuals is used initially. The reference point $z^\star$ is updated online to the best-encountered objective values. Parameters δ and nr are set respectively to 0.1 and 2 in MOEA/D$_{(\delta, nr)}$ [8]. We follow the performance assessment design proposed in [7] using the hypervolume (to be maximized) and the additive epsilon (to be minimized) indicators [22]. For each instance, we compute a reference set by merging the approximation sets over all runs. The reference point is set to the worst value in each dimension of the reference set.

4.2 Overall Performance

We start our empirical study by providing in Table 1 a global overview of the relative performance of SET-MOEA/D and MOEA/D. Out of the 72 settings (12 ρMNK-landscapes $\times$ 3 μ values $\times$ 2 indicators), SET-MOEA/D is in average better than MOEA/D 62 times (i.e. more than 85% of the cases). SET-MOEA/D significantly outperforms MOEA/D 26 times (36%), whereas MOEA/D significantly outperforms

Table 1: Average and standard deviation (in braces) of the indicator-values achieved by SET-MOEA/D and MOEA/D at 10^6 evaluations. For every (μ, ρ, M), an *underlined* font indicates that the algorithm in the column is better (w.r.t. considered indicator). When the difference is statistically significant according to a Wilcoxon test (p-value = 0.05), a *bold* font is additionally used.

		$M = 2$		$M = 3$		$M = 4$		$M = 5$	
		SET-MOEA/D	MOEA/D	SET-MOEA/D	MOEA/D	SET-MOEA/D	MOEA/D	SET-MOEA/D	MOEA/D
		HYPERVOLUME INDICATOR ($\times 10^3$)							
$\mu = 100$	$\rho = 0.2$	$\underline{31.088}_{(1.14)}$	$30.900_{(1.02)}$	$\underline{5.738}_{(0.22)}$	$5.678_{(0.20)}$	$\underline{0.663}_{(0.05)}$	$0.657_{(0.06)}$	$0.090_{(0.01)}$	$\underline{0.094}_{(0.01)}$
	$\rho = 0$	$\underline{28.653}_{(0.72)}$	$28.286_{(0.75)}$	$\underline{\mathbf{9.863}}_{(0.12)}$	$9.774_{(0.18)}$	$\underline{2.308}_{(0.05)}$	$2.295_{(0.06)}$	$\underline{0.491}_{(0.02)}$	$0.490_{(0.03)}$
	$\rho = -0.2$	$\underline{46.941}_{(0.95)}$	$46.878_{(1.01)}$	$21.539_{(0.52)}$	$\underline{21.580}_{(0.32)}$	$7.664_{(0.22)}$	$\underline{7.687}_{(0.22)}$	$1.568_{(0.04)}$	$\underline{1.572}_{(0.04)}$
$\mu = 200$	$\rho = 0.2$	$\underline{31.102}_{(1.02)}$	$30.948_{(0.73)}$	$\underline{\mathbf{6.117}}_{(0.13)}$	$5.947_{(0.19)}$	$\underline{0.749}_{(0.04)}$	$0.730_{(0.05)}$	$0.106_{(0.01)}$	$\underline{0.108}_{(0.01)}$
	$\rho = 0$	$\underline{28.721}_{(0.68)}$	$28.574_{(0.63)}$	$\underline{10.166}_{(0.13)}$	$10.102_{(0.14)}$	$\underline{\mathbf{2.488}}_{(0.05)}$	$2.454_{(0.06)}$	$\underline{\mathbf{0.581}}_{(0.03)}$	$0.563_{(0.02)}$
	$\rho = -0.2$	$\underline{47.152}_{(0.92)}$	$47.101_{(0.82)}$	$\underline{22.530}_{(0.38)}$	$22.303_{(0.43)}$	$\underline{8.429}_{(0.12)}$	$8.374_{(0.13)}$	$\underline{\mathbf{1.866}}_{(0.03)}$	$1.817_{(0.03)}$
$\mu = 600$	$\rho = 0.2$	$\underline{31.362}_{(0.95)}$	$31.157_{(0.58)}$	$\underline{6.225}_{(0.11)}$	$6.162_{(0.18)}$	$\underline{\mathbf{0.818}}_{(0.04)}$	$0.774_{(0.06)}$	$\underline{\mathbf{0.130}}_{(0.01)}$	$0.117_{(0.01)}$
	$\rho = 0$	$\underline{28.814}_{(0.46)}$	$28.344_{(0.79)}$	$\underline{\mathbf{10.497}}_{(0.09)}$	$10.380_{(0.12)}$	$\underline{\mathbf{2.690}}_{(0.04)}$	$2.601_{(0.07)}$	$\underline{\mathbf{0.675}}_{(0.02)}$	$0.640_{(0.02)}$
	$\rho = -0.2$	$\underline{47.459}_{(0.71)}$	$47.394_{(0.77)}$	$\underline{\mathbf{23.517}}_{(0.52)}$	$23.049_{(0.39)}$	$\underline{\mathbf{9.369}}_{(0.15)}$	$9.095_{(0.14)}$	$\underline{\mathbf{2.241}}_{(0.04)}$	$2.156_{(0.05)}$
		EPSILON INDICATOR ($\times 10^3$)							
$\mu = 100$	$\rho = 0.2$	$\underline{16.758}_{(5.09)}$	$17.981_{(4.32)}$	$\underline{29.163}_{(5.37)}$	$30.085_{(4.14)}$	$\underline{44.384}_{(5.35)}$	$47.798_{(6.86)}$	$\underline{53.595}_{(7.73)}$	$53.696_{(6.50)}$
	$\rho = 0$	$\underline{13.545}_{(6.36)}$	$14.262_{(5.57)}$	$\underline{25.936}_{(4.08)}$	$27.280_{(2.82)}$	$\underline{45.575}_{(3.78)}$	$46.507_{(5.25)}$	$70.286_{(4.25)}$	$\underline{\mathbf{65.958}}_{(5.37)}$
	$\rho = -0.2$	$\underline{17.915}_{(5.59)}$	$19.255_{(7.44)}$	$\underline{42.001}_{(5.82)}$	$42.531_{(5.51)}$	$73.402_{(3.20)}$	$\underline{70.631}_{(4.04)}$	$94.785_{(4.21)}$	$\underline{93.876}_{(3.85)}$
$\mu = 200$	$\rho = 0.2$	$\underline{18.017}_{(4.63)}$	$19.872_{(4.48)}$	$\underline{\mathbf{23.128}}_{(6.85)}$	$28.168_{(6.92)}$	$\underline{39.180}_{(5.07)}$	$40.534_{(5.59)}$	$\underline{\mathbf{49.246}}_{(7.08)}$	$56.998_{(10.93)}$
	$\rho = 0$	$\underline{12.191}_{(4.34)}$	$13.559_{(5.70)}$	$\underline{\mathbf{21.908}}_{(2.80)}$	$26.148_{(3.75)}$	$\underline{41.249}_{(3.68)}$	$41.990_{(5.66)}$	$\underline{57.324}_{(4.93)}$	$60.197_{(6.31)}$
	$\rho = -0.2$	$\underline{19.636}_{(5.33)}$	$19.722_{(6.01)}$	$\underline{38.930}_{(10.50)}$	$45.049_{(11.62)}$	$62.758_{(1.66)}$	$\underline{\mathbf{60.921}}_{(4.43)}$	$77.930_{(3.22)}$	$\underline{76.383}_{(3.40)}$
$\mu = 600$	$\rho = 0.2$	$\underline{16.206}_{(8.21)}$	$17.628_{(4.39)}$	$\underline{\mathbf{23.156}}_{(5.86)}$	$29.094_{(7.35)}$	$\underline{\mathbf{35.187}}_{(5.73)}$	$42.370_{(7.40)}$	$\underline{\mathbf{40.739}}_{(8.48)}$	$56.427_{(10.79)}$
	$\rho = 0$	$\underline{\mathbf{10.927}}_{(4.19)}$	$15.952_{(6.46)}$	$\underline{\mathbf{21.105}}_{(3.33)}$	$25.717_{(4.29)}$	$\underline{\mathbf{35.436}}_{(4.96)}$	$44.992_{(4.75)}$	$\underline{\mathbf{46.554}}_{(4.59)}$	$60.742_{(8.33)}$
	$\rho = -0.2$	$\underline{16.656}_{(5.03)}$	$18.086_{(5.81)}$	$\underline{\mathbf{36.026}}_{(13.32)}$	$48.277_{(11.54)}$	$\underline{\mathbf{50.600}}_{(4.82)}$	$55.374_{(5.47)}$	$\underline{\mathbf{61.547}}_{(4.31)}$	$69.937_{(7.06)}$

SET-MOEA/D only 2 times (3%), and there is no statistically significant difference 44 times (60%). Actually, a similar overall superiority of SET-MOEA/D is found when compared to MOEA/D$_{(\delta,nr)}$, but this is not reported in Table 1 due to space restrictions. Notice that SET-MOEA/D is never statistically outperformed w.r.t hypervolume.

Furthermore, we can observe a notable change in the relative performance of the two algorithms when looking more carefully at dimension M. Although there is no statistical difference for *bi*-objective problem instances (except for $\rho = 0$ and $\mu = 600$ where SET-MOEA/D is better in terms of epsilon indicator), SET-MOEA/D is consistently at least as good as MOEA/D, and often significantly better, when the number of objectives increases $M \geqslant 3$ (except for 2 particular cases, w.r.t the epsilon indicator). Such an observation indicates that SET-MOEA/D is a viable alternative to MOEA/D for multi- and many-objective optimization. Interestingly, SET-MOEA/D is always outperforming MOEA/D when using the largest population size ($\mu = 600$), which is a particularly relevant feature for many-objective optimization. This is confirmed by the following anytime analysis.

4.3 Anytime Behavior

When studying the anytime performance of the considered algorithms, the superiority of SET-MOEA/D is even more pronounced when the maximum size allowed for the population set is sufficiently large and using a reasonable budget in term of function evaluations. In fact, we depict in Fig. 2 the evolution of the average epsilon indicator value as a function of the number of function evaluation calls, while additionally including in our comparison the extended variant MOEA/D$_{(\delta,nr)}$, for $\mu = 600$. One can clearly see that SET-MOEA/D has a better anytime convergence profile than both MOEA/D and MOEA/D$_{(\delta,nr)}$ for multi- and many-objective problems ($M \in \{3, 4, 5\}$), and especially for $M = 5$. Actually, the results are mitigated for bi-objective problem instances ($M = 2$), especially starting at 10^5 evaluations, but before that SET-MOEA/D is the best-performing approach. This is a desirable feature especially when dealing with an expensive optimization scenario when one is only allowed a restricted number of function evaluation calls.

These findings hold independently of the objective correlation ρ. This is a notable feature since one might typically expect that the impact of different ρ-values on the landscape, in terms of solution diversity and Pareto front size, could influence the diversity-to-convergence trade-off of the experimented algorithms. Furthermore, we can extract an interesting observation when analyzing more carefully the anytime behavior of MOEA/D$_{(\delta,nr)}$. At the early stages of the search process, MOEA/D$_{(\delta,nr)}$ has the slowest convergence behavior, whereas SET-MOEA/D is the most efficient. This can be attributed to the specific replacement design of MOEA/D$_{(\delta,nr)}$ which has a tendency to preserve solution diversity. As the search progresses, MOEA/D$_{(\delta,nr)}$ is overall able to outperform MOEA/D for $M \in \{2, 3\}$ but to a less extent for many-objective problem instances $M \in \{3, 4, 5\}$, while being always worse than SET-MOEA/D.

In Fig. 3, we further illustrate the difference in the anytime behavior of the three algorithms w.r.t. the different values of μ (due to space restrictions, only one many-objective instance is shown). Two insightful observations can be extracted. First, SET-MOEA/D appears to be relatively robust to the choice of the cardinality bound μ of the solution set. Using different μ-values leads to different indicator values. With no surprise, larger μ-values lead to better

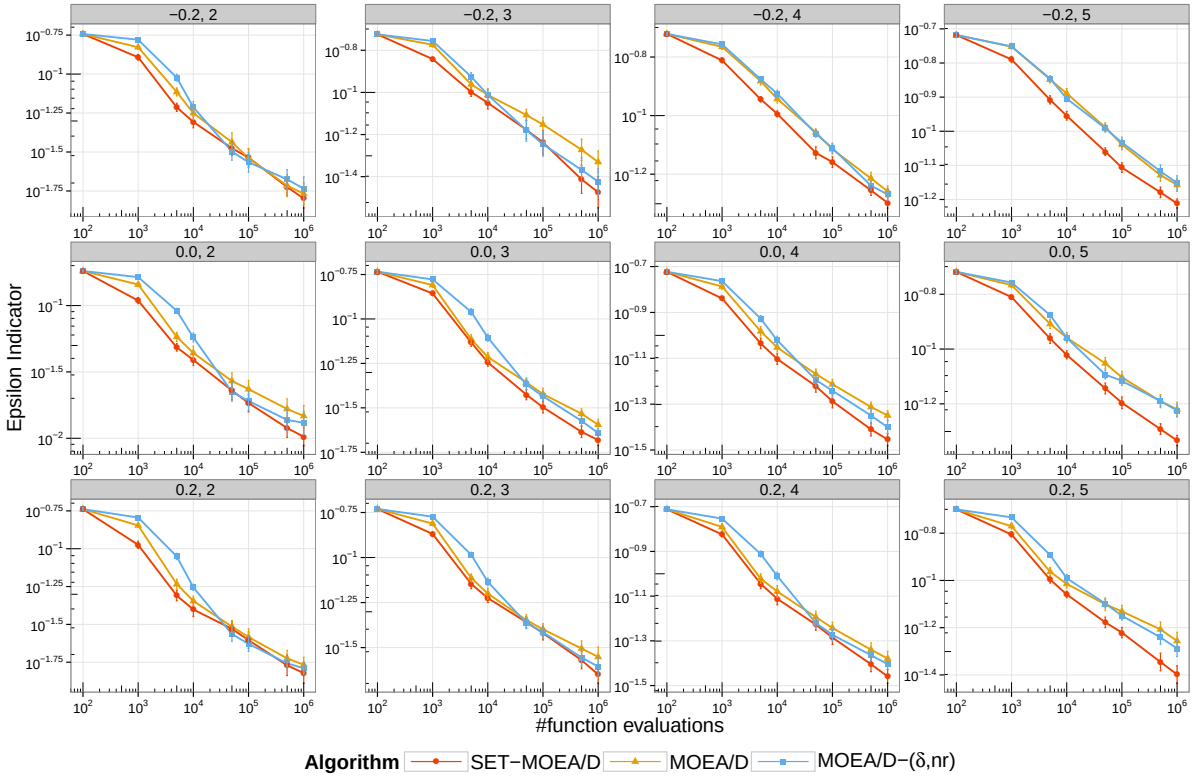

Figure 2: Anytime performance of SET-MOEA/D, MOEA/D, and MOEA/D$_{(\delta,nr)}$ with a (maximum) population size $\mu = 600$. Subfigures in rows and columns correspond to instances with respectively the different objective correlation $\rho \in \{-0.2, 0.0, 0.2\}$ and the different number of objectives $M \in \{2, 3, 4, 5\}$. Notice the log-scales. Error bars show 95% confidence intervals.

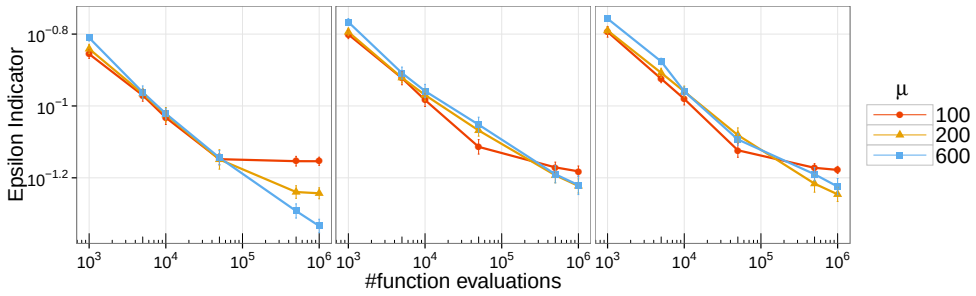

Figure 3: Anytime performance of SET-MOEA/D (left), MOEA/D (middle), MOEA/D$_{(\delta,nr)}$ (right); using different values of (maximum) population size μ for an instance with $\rho = 0.0$ and $M = 5$. Error bars show 95% confidence intervals.

performance. However, for MOEA/D and MOEA/D$_{(\delta,nr)}$, this is only true at the late stages of the search, whereas this tends to hold at anytime for SET-MOEA/D (except with very few function evaluations). At the beginning of the run, the cardinality bound of the set maintained by SET-MOEA/D has almost no impact on the algorithm performance. Only at the end of the run, having a larger population size allows for more improvement, especially for many-objective problems. This difference implies that the performance of MOEA/D and MOEA/D$_{(\delta,nr)}$ is quite sensitive to the setting of the population size μ, in the sense that it is correlated to *both* the available budget and to μ. In contrast, in the case of SET-MOEA/D, performance seems to be simply positively correlated to the available budget, which is a natural outcome, especially for many-objective problems. We attribute this feature to the fact that parameter μ plays a fundamentally different role in the design of SET-MOEA/D. In fact, parameter μ is to be viewed as *one* way to structure the population set while imposing a bound on its cardinality. This is to be contrasted to the initial design of MOEA/D, where it also influences the intrinsic composition of the population, thus impacting more critically the search behavior, e.g., by introducing duplicates. This is analyzed further in the rest of the paper.

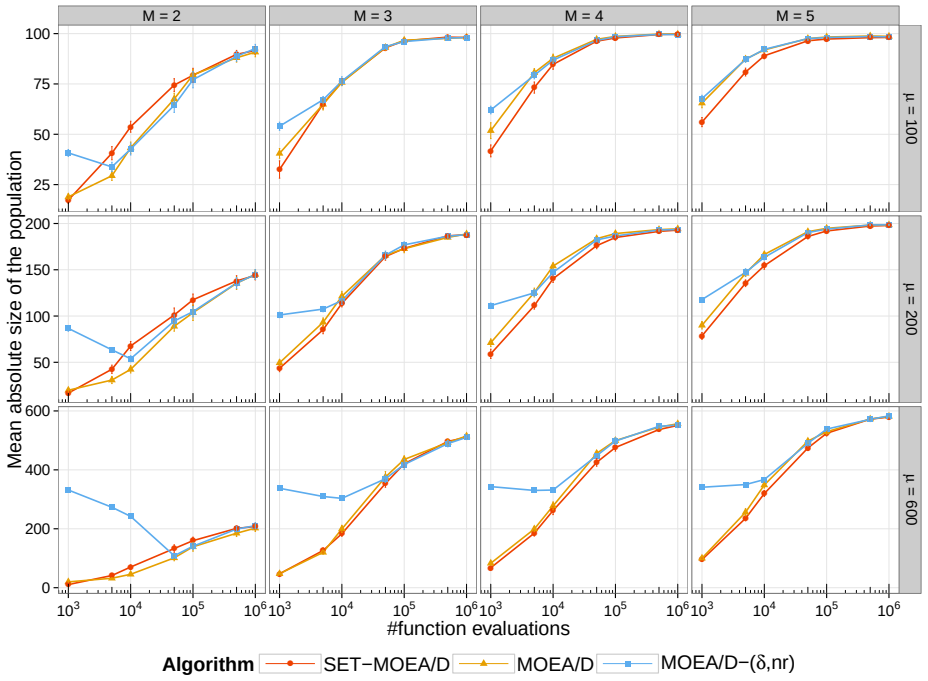

Figure 4: Evolution of the average absolute size of the population for the different algorithms for different $\mu \in \{100, 200, 600\}$ (in rows), and for the instances with $M \in \{2, 3, 4, 5\}$ (in columns) and $\rho = 0.0$. Error bars show 95% confidence intervals.

4.4 Population Dynamics

As discussed previously, unlike in Set-Moea/d, some duplicated solutions can be maintained by Moea/d. In Fig. 4, we show the evolution of the *absolute* (average) size of the population maintained by the different considered algorithms. In the case of Moea/d and Moea/d$_{(\delta,nr)}$, we simply count each single objective vector only once (since there are no equivalent solutions in ρMNK-landscapes).

(Average) Absolute Population Size. The long-term general tendency is roughly the same for all three algorithms. The mean absolute size of the population starts at a small value within relatively few function evaluations (10^3), then it grows as the search process evolves. The maximum-allowed size μ can only be reached in the setting where the number of objectives *and* the computing budget are both relatively high. For a given budget (e.g., 10^5), using more weight vectors (e.g., 200) allows the set to attain an even larger size (e.g., > 150) than the maximum allowed when using fewer weight vectors (e.g., 100). Based on our anytime analysis, this does not necessarily implies that the performance, w.r.t. some budget, is better when using more weight vectors because the population is more diverse. This actually renders a more complex behavior that can only be understood by taking into account the mapping of the weight vectors into the Pareto front. In fact, fewer weight vectors, i.e., smaller μ-values, are likely to imply more intensification, which may in turn make the search stuck more easily.

Additionally, we can observe that the absolute size of the population tends to be slightly lower in Set-Moea/d at the early stage of the search process, with the exception of bi-objective instances. The dynamics of Moea/d$_{(\delta,nr)}$ is also different at the beginning, because the number of replacements is restricted to at most nr. However, even if the propagation of duplicates is slower, Moea/d$_{(\delta,nr)}$ does *not* properly address the issue induced by having duplicates in the population in the long run, and the absolute size of the population drops to the same level than the other algorithms after about 100 generations. As a consequence, we argue that the overall good performance of Set-Moea/d is not only due to the absolute number of solutions (which in fact was shown to be roughly the same). This should make it clear that adopting a set-oriented design allows us to have a better control on the population. For instance, the selection of duplicates can typically have a non-zero probability in Moea/d. By contrast, we are not only able to avoid such a bias, but we can also refine the selection by controlling explicitly which individuals to choose, i.e. through the neighborhood relation $\mathcal{BS}$.

Population Behavior. We can refine our understanding of the population dynamics by analyzing not only its *average* absolute size, but its *actual* absolute size in *one single* particular execution. This is depicted in Fig. 5, corresponding to a typical run of Set-Moea/d. We can see that the absolute population size is indeed increasing overall, but in a non-smooth manner. In fact, the population size decreases periodically in an abrupt manner, and then increases again to reach an even higher value. This can be explained as follows. The algorithm operates in several periods. At each period, newly-found, hopefully diverse, solutions are produced and mapped to different weight vectors. This hence helps the algorithm to jump to an even more efficient region of the objective space, which makes the population size decrease again, an so on. The periodicity of this behavior is however decreasing with time (notice the log-scale in Fig. 5), which can indicate that the algorithm is actually converging.

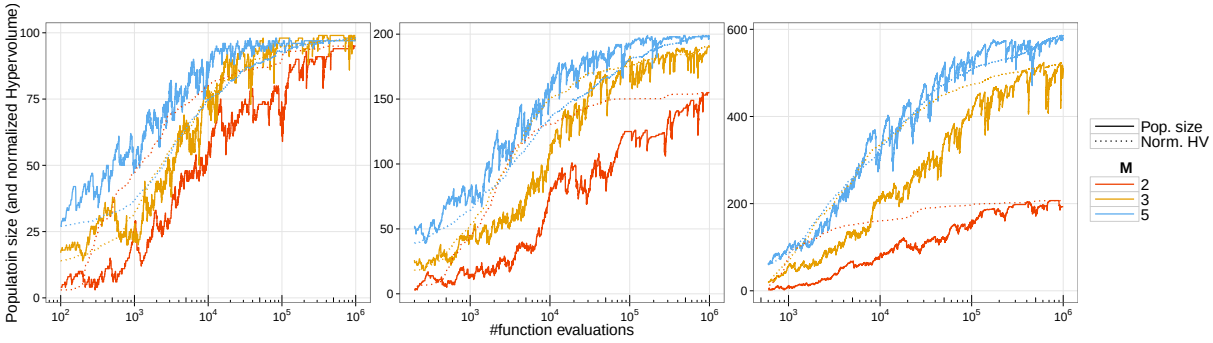

Figure 5: Dynamics of the population size in a typical run of SET-MOEA/D on instances with $M \in \{2, 3, 5\}$ ($M = 4$ is omitted for clarity) and $\rho = 0.0$. The left, middle, and right subfigures correspond to a μ-value of 100, 200 and 600, respectively. The Hypervolume (in dashed font) is normalized w.r.t. the max and min values attained by the population size to better visualize any possible correlation. Only the Hypervolume evolution is shown, but not its scale, for more clarity.

Interestingly, such a dynamic implies a smooth evolution of the population quality, which suggests that the population jump is periodically localized in small regions of the objective space.

5 CONCLUSION AND PERSPECTIVES

By adopting a a set-oriented perspective, the work conducted in this paper enables to provide an alternative design of multi-objective decomposition-based algorithms. The proposed SET-MOEA/D algorithm follows a standard evolutionary search process using a many-to-one mapping between weight vectors and solutions in order to perform selection and replacement. Besides being able to improve the quality of the approximation sets, our experimental analysis provides insights into the behavior of the underlying evolutionary process, while highlighting some interesting features that need to be studied in a more systematic manner in the future. For instance, it would be nice to provide decomposition-based features to better estimate the diversity of the population set, as well as its degree of convergence. This could be helpful to control selection and replacement in an adaptive manner, or to adjust the maximum size of the population online, which we believe to be a challenging but crucially-important perspective. In our design, the replacement is relatively aggressive, since poor quality solutions die immediately. This could be handled differently in the future if diversity appears to be an issue. Moreover, we only consider combinatorial instances in our experiments. It would be interesting to confirm the benefits of such an alternative set-oriented design when tackling other instances from both discrete and continuous domains. Interestingly, the design of the proposed algorithm is flexible enough to enable the integration of other advanced EMO techniques, especially when dealing with many-objective optimization problems, for which a set-oriented perspective seems to be particularly promising.

Acknowledgments. This work was supported by the French national research agency (ANR-16-CE23-0013-01) and the Research Grants Council of the Hong Kong Special Administrative Region, China (RGC Project No. A-CityU101/16).

REFERENCES

[1] H. E. Aguirre and Kiyoshi Tanaka. 2007. Working principles, behavior, and performance of MOEAs on MNK-landscapes. *EJOR* 181, 3 (2007), 1670–1690.
[2] Nicola Beume, Boris Naujoks, and Michael Emmerich. 2007. SMS-EMOA: Multiobjective selection based on dominated hypervolume. *European Journal of Operational Research* 181, 3 (2007), 1653–1669.
[3] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan. 2002. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation* 6, 2 (2002), 182–197.
[4] Ioannis Giagkiozis, Robin C. Purshouse, and Peter J. Fleming. 2013. Generalized Decomposition. In *Evolutionary Multi-Criterion Optimization*. 428–442.
[5] E. J. Hughes. 2003. Multiple Single Objective Pareto Sampling. In *CEC*. 2678–2684.
[6] S. A. Kauffman. 1993. *The Origins of Order*. Oxford University Press.
[7] J. Knowles, L. Thiele, and E. Zitzler. 2006. *A Tutorial on the Performance Assessment of Stochastic Multiobjective Optimizers*. TIK Report 214. Computer Engineering and Networks Laboratory (TIK), ETH Zurich, Zurich, Switzerland.
[8] H. Li and Q. Zhang. 2009. Multiobjective Optimization Problems With Complicated Pareto Sets, MOEA/D and NSGA-II. *IEEE TEC* 13, 2 (2009), 284–302.
[9] K. Li, Q. Zhang, S. Kwong, M. Li, and R. Wang. 2014. Stable Matching-Based Selection in Evolutionary Multiobjective Optimization. *IEEE TEC* 18, 6 (2014), 909–923.
[10] Gauvain Marquet, Bilel Derbel, Arnaud Liefooghe, and El-Ghazali Talbi. 2014. Shake Them All! Rethinking Selection and Replacement in MOEA/D. In *Parallel Problem Solving from Nature-PPSN XIII*. 641–651.
[11] Saúl Zapotecas Martínez, Hernán E. Aguirre, Kiyoshi Tanaka, and Carlos A. Coello Coello. 2015. On the low-discrepancy sequences and their use in MOEA/D for high-dimensional objective spaces. In *CEC*. 2835–2842.
[12] K. Miettinen. 1999. *Nonlinear Multiobjective Optimization*. Kluwer.
[13] Hugo Monzón, Hernán Aguirre, Sébastien Verel, Arnaud Liefooghe, Bilel Derbel, and Kiyoshi Tanaka. 2017. Closed State Model for Understanding the Dynamics of MOEAs. In *GECCO*. 609–616.
[14] Tadahiko Murata, Hisao Ishibuchi, and Mitsuo Gen. 2000. Cellular Genetic Local Search for Multi-objective Optimization. In *GECCO*. 307–314.
[15] L. Paquete, M. Chiarandini, and T. Stützle. 2004. Pareto Local Optimum Sets in the Biobjective Traveling Salesman Problem: An Experimental Study. In *Metaheuristics for Multiobjective Optimization*. Chapter 7, 177–199.
[16] A. Trivedi, D. Srinivasan, K. Sanyal, and A. Ghosh. 2017. A Survey of Multiobjective Evolutionary Algorithms Based on Decomposition. *IEEE TEC* 21, 3 (2017), 440–462.
[17] Sébastien Verel, Arnaud Liefooghe, Laetitia Jourdan, and Clarisse Dhaenens. 2013. On the structure of multiobjective combinatorial search space: MNK-landscapes with correlated objectives. *EJOR* 227, 2 (2013), 331–342.
[18] Z. Wang, Q. Zhang, A. Zhou, M. Gong, and L. Jiao. 2016. Adaptive Replacement Strategies for MOEA/D. *IEEE Transactions on Cybernetics* 46, 2 (2016), 474–486.
[19] Qingfu Zhang and Hui Li. 2007. MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition. *IEEE TEC* 11, 6 (2007), 712–731.
[20] Eckart Zitzler and Simon Künzli. 2004. Indicator-based selection in multiobjective search. In *Parallel Problem Solving from Nature-PPSN VIII*. 832–842.
[21] E. Zitzler, L. Thiele, and J. Bader. 2010. On Set-Based Multiobjective Optimization. *IEEE TEC* 14, 1 (2010), 58–79.
[22] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. Grunert da Fonseca. 2003. Performance Assessment of Multiobjective Optimizers: An Analysis and Review. *IEEE TEC* 7, 2 (2003), 117–132.